Defense Information Infrastructure
Common Operating Environment(DII-COE)

Distributed Computing

Software Requirements Specification (SRS)

9 July 1997

Changes since 28 January 1997 draft:

Strikeouts are in blue text, new text is in red. Green text should be revisited.

Major changes - added requirements traceability matrix to section 5; tentatively included requirements received from AF as part of COE 3.2 requirements call.

1. Title: Changed the title to reflect that the SRS is a cumulative requirements document, not specific to COE 4.0.

2. Section 2: updated several references.

3. Section 3.2.1.1.3: Added firewall requirement for simultaneous operation in 2 cells from Army. (later deleted after discussion with Army). Added note to indicate that DCWG should develop guidance on how to configure the system to enable distributed computing to work through firewalls. Added note that security TWG is not recommending a single firewall, so the distributed computing implementation must work with multiple different firewall products.

4. Section 3.2.1.6.1: Added Army requirement for Rational Apex compiler support for Ada. (later deleted after discussion with Army)

5. Section 3.2.1.6.3 Added Compiler support requirement.

6. Sections 3.2.1.10-12 Removed because requirements for segmentation, standards, and platforms are not specific to distributed computing and should be covered in a blanket requirements document.

7. Section 3.2.1.13-14: Removed because legacy compatibility and product quality are evaluation criteria more than requirements (moved to appendix c).

8. Section 3.2.1.15 Removed Training, since it is already covered in 3.14.

9. Section 3.2.1.17 Added requirement for Y2K compliance.

10. Section 3.2.2.4.3: Requirement for C++ interface to DCE and GSS-API removed based on lack of commercial support.

11. Section 3.2.2.4.4-5: Requirement for DCE traffic monitor/debugger moved to section 8, recommended for inclusion in the COE Developer’s Toolkit.

12. Section 3.2.2.4.5: Requirement for DCE templates and DCE wrappers moved to section 8, recommended for inclusion in COE Developer’s Toolkit.

13. Section 3.2.2.11: Added this section to accomodate Army requirement for create and maintain directory of workstations scheduled to be in contact with cell.

14. Section 3.2.2.12: Added section on DCE Bindings.

15. Section 3.2.2.12.1: Added requirement for Ada/DCE bindings using Rational Apex compiler. (later deleted based on discussion with Army)

16. Section 3.2.3.1 (CORBA Background) removed.

17. Section 3.2.3.4: Added note about the viability of OpenDoc based technology as a CORBAfacility.

18. Section 3.2.3.6.1-2: Requirements for inter-orb traffic monitor/debugger and development templates moved to section 8, recommended for inclusion in the COE Developer's Toolkit.

19. Section 5: Added the requirements traceability matrix done in March 1997.

20. Section 5: Tentatively moved the note about a unique Navy requirement not being fulfilled by DCWG recommended queueing product into the notes column of the requirements traceability matrix for requirement 3.2.8.13 (LIFO).

21. Section 7.1.6: Tentatively added recommendation for Orbix/Ada. Need to discuss at next DCWG.

22. Section 8.2.1.1.4: Added single login requirement from LCM.

23. Section 8.8.1.1.1: Added requirement for CORBA compliance web server.

24. Section 8.8.1.1.2: Added requirement for DCE newsgroup authentication.

25. Section 8.7: Added section for Data Access Services

26. Section 8.7.1.1.1-3: Added Data Access Requirements

SECTION 1

SCOPE

1.1 IDENTIFICATION

This specification describes the requirements for the distributed computing services and their interfacing with other functional elements of the Defense Information Infrastructure (DII) Common Operating Environment (COE). The distributed computing component and its relationship to the rest of the COE, is described in the COE Baseline Description Document.

1.2 SYSTEM OVERVIEW

The focus of the COE's distributed computing component is on distributed computing capabilities that permit procedures and objects to be invoked on remote hosts as though they were local to the calling module. In addition to these basic capabilities, the distributed computing component will include a variety of enabling services, such as security, time, persistence, and naming; many of these services are required for the development of applications that are distributed. The two fundamental technologies that will be implemented in the COE are the Distributed Computing Environment (DCE) and the Common Object Request Broker Architecture (CORBA), including some related services; These technology choices are based on requirements from Department of Defense (DoD) services and related agencies.

This Software Requirements Specification (SRS) focuses on specifying requirements for the implementation of these basic technologies in the COE, as well as for related capability requirements that may not be addressed by those two basic technologies. Related capability requirements may include requirements relating to the integration of the distributed computing component with other components or capabilities in the COE.

1.3 DOCUMENT OVERVIEW

NOTE: The requirements specified herein assume a familiarity with the concepts of distributed computing and with the two specific technologies, DCE and CORBA, that are being used to implement the distributed computing component of the COE.

Section 2 specifies documents that are referenced elsewhere in this SRS.

Section 3 specifies requirements for the COE distributed computing component, with the bulk of the requirements being specified in section 3.2. Section 3.2 is subdivided as follows:

Section 3.2.1 specifies fundamental or common requirements for distributing computing which are not specific to either of the two technologies that are being focused upon.

Section 3.2.2 specifies requirements that are specific to DCE.

Section 3.2.3 specifies requirements that are specific to CORBA.

Section 4 is describes qualification provisions.

Section 5 provides a requirements traceability table that identifies the source of requirements.

Section 6 contains miscellaneous notes.

Appendix A describes recommendations that the COE Distributed Computing Technical Working Group has made to the DII COE Architecture Oversight Group (AOG).

Appendix B describes capabilities that are outside the scope of the distributed computing component, and are recommended to be included in the requirements specifications for other COE components.

SECTION 2

REFERENCED DOCUMENTS

Several of the documents listed below are evolving. The most recent version of these documents are listed, but this list should be periodically updated to track the evolution of those documents.

2.1 GOVERNMENT DOCUMENTS

1. GCCS Baseline Common Operating Environment, November 28, 1994
2. GCCS Integration Standard version 2.0, October 23, 1995
3. User Interface Specifications for the Defense Information Infrastructure (DII), Draft Version 2.0, April 1996
4. DRAFT, Architectural Design Document for the Global Command and Control System (GCCS) Common Operating Environment (COE), DISA, July 24, 1995
5. "GCCS Implementation of the Distributed Computing Environment Version 1.0", DISA, September 1995.
6. Defense Information Infrastructure (DII) Common Operating Environment (COE) Integration and Runtime Specification (I&RTS), Version 2.0, 23 October, 1995

2.2 NON-GOVERNMENT DOCUMENTS

NOTE: The following list of references includes documents from the Object Management Group (OMG) that are available either in a combined CORBAservices specification, or separately as OMG documents. Both references are given and the content should be identical in either the individual or combined formats. In the event of a discrepancy, the OMG should be consulted to determine which document is in error.

NOTE: Most, if not all of these services will be purchased as COTS for use within the COE, such that specification issues are unlikely to be an issue for COE developers.

1. OSF DCE Application Development Guide, Revision 1.0, Prentice Hall, 1993.
2. OSF DCE Application Development Reference, Revision 1.0, Prentice Hall, 1993.
3. Rosenberry, W., D. Kenney, and G. Fisher, Understanding DCE, O’Reilly & Associates, 1992.

4. Shirley, J., W. Hu, and D. Magid, Guide to Writing DCE Applications, Second Edition, O'Reilly & Associates, 1992.

5. Thompson, J. and Otto, E. "Distributed Computing Environment(DCE) Lessons Learned," Logicon, July 21, 1995

6. Common Facilities Architecture, Revision 3.0, OMG Document 94-11-9, The Object Management Group, Framingham, MA, November 14, 1994.

7. Common Object Request Broker: Architecture and Specification, Revision 1.2, OMG Document 93-12-43, The Object Management Group, Framingham, MA.

8. Common Object Services Specification, Volume I, Revision 1.0, First Edition, OMG Document 94-1-1, The Object Management Group, Framingham, MA, March 1, 1994.

9. Object Management Architecture Guide, Revision 2.0, Second Edition, OMG TC Document 92-11-1, The Object Management Group, Framingham, MA, September 1, 1992.

10. Object Services Architecture, Revision 8.0, OMG Document 94-11-12, The Object Management Group, Framingham, MA, December 9, 1994.

11. The Common Object Request Broker: Architecture and Specification, Document Number Formal/97-02-25

12. C++ Language Mapping 1.1, CORBA 2.0/IIOP specification, Document tc/96-01-13, 19 Mar, 1996

13. Ada Language Mapping, Document 1995/95-05-16, 19 Mar, 1996

14. Smalltalk Language Mapping (See OMG CORBA 2.0/IIOP web page), Document 1994/94-11-08, 28 Mar, 1995

15. CORBA Interoperability (Revised CORBA Interoperability Specification), Document interop/96-05-01, 30 Oct,1996

16. Common Secure IIOP (CSI), Document orbos/96-06-20, 11 Mar, 1997

17. IDL COBOL Mapping, Document orbos/96-10-01, 11 Mar, 1997

18. IDL Fixed Point Extensions, Document orbos/96-05-04, 20 Aug,1996

19. IDL Type Extensions (IDL Type Extensions for decimal data types), Document orbos/96-05-04, 30 Oct, 1996

20. CORBAServices: Common Object Services Specification (contains the following services: Event Notification, Lifecycle, Naming, Persistent Object, Relationship, Object Transaction, Concurrency Control, Externalization, Object Query, Licensing, Properties, Object Security, Time), Document Number Formal/97-02-24

21. Object Collections Service, Document orbos/96-05-05, 30 Oct, 1996

22. Object Trader Service, Document orbos/96-05-06, 30 Oct, 1996, (also Document number orbos/96-06-07)

23. CORBAfacilities: Common Facilities Architecture, document number not defined yet by OMG

24. Compound Presentation & Interchange, Documents 95-12-30 through 95-12-34 (Distributed Document Component Facility), 19 Mar, 1996

25. System Management Facility, Document numbers 1995/95-12-02 through 1995/95-12-06, Document 1995/95-12-02, 21 Nov, 1996

SECTION 3

REQUIREMENTS

3.1 REQUIRED STATES AND MODES

The required modes and states of the system are assumed to be documented once for the entire COE, and therefore is not discussed herein, since this SRS focuses only on the distributed computing component of the COE.

3.2 CSCI CAPABILITY REQUIREMENTS

As mentioned earlier, the distributed computing component of the COE supports the two fundamental, industry standard technologies for distributed computing, DCE and CORBA. DCE supports a *remote procedure call* (RPC) paradigm of software development, whereas CORBA supports a *distributed object management* (DOM) paradigm. There are, however, requirements that are fundamental or at a higher level than either DCE or CORBA, or which are common to both paradigms, which are specified in Section 3.2.1. DCE specific requirements are specified in Section 3.2.2, and CORBA specific requirements are specified in Section 3.2.3, below.

Some fundamental requirements for items like Remote Procedure Call or Object Request Broker functionality are not specified herein since such requirements are inherent in the DCE ad CORBA technologies that have been selected for implementation in the COE.

NOTE: in the following material, the word "implementation" is used to refer to the general set of combined capabilities used to implement the distributed computing requirements. The implementation may include both COTS and GOTS components. Additionally, the word "participant" is used to refer to entities that make use of the distributed computing implementation (in DCE terminology, these are generally referred to as principals). The word "domain" is used to generally refer to a DCE cell or a CORBA namespace, which typically correspond to a system management or security scope.

3.2.1 Fundamental or Common Requirements

The requirements specified in this section of the SRS are fundamental or common to the implementation of distributed computing in the COE, and apply to the implementation of both of the DCE and CORBA technologies (including related infrastructure services) in the COE.

3.2.1.1 Security

3.2.1.1.1 General Security Requirements. The implementation shall comply with, or support, those security requirements specified in the Security SRS for the COE that are applicable to the distributed computing component, including at least the following:

a) Mandatory access control [TBD]

b) Discretionary access control

c) Mutual Identification and Authentication

d) Authorization

e) Privacy

f) Integrity

g) Non-Repudiation

h) Auditing

3.2.1.1.2 Fortezza Integration. The implementation shall provide a FORTEZZA/MISSI compliant alternative encryption mechanism that is usable by all of the applicable distributed computing services.

3.2.1.1.3 Firewalls. The implementation shall provide support for use through firewalls and guards.

NOTE: Operation of the implementation through firewalls and guards is probably not a requirement that can be directly satisfied by the distributed computing component of the COE, but is more likely to be satisfied through the configuration of the entire system/network, including routers, packet filtering, intermediate hosts, etc. Even so, the implementation of distributed computing shouldn't deny service in such configurations. The DCWG should develop general guidance on how to configure the system to enable distributed computing to work through firewalls.

NOTE: The Security Working Group is not recommending a particular firewall, so any guidance will be generic. The distributed computing implementation may have to work with many different firewall products NRO has identified a need for Orbix/IIOP to work through firewalls.

3.2.1.2 Dynamic Reconfigurability

3.2.1.2.1 Location Independence. The implementation shall be able to determine the location of resources by using a location-independent name to permit a client participant to bind to a resource regardless of its physical location (e.g., to support relocation of services to another host).

3.2.1.2.2 Replication. The implementation shall provide the ability to replicate its own servers (i.e., those that support the distributed computing implementation itself) and application-level services (e.g., a map server, correlation server, etc) to support fault tolerant operation and optimal performance.

3.2.1.2.3 Flexible Domain Configuration. The implementation shall have the flexibility to support a variety of deployment options, including the ability to subdivide a domain (e.g., cell or namespace) and incorporate the subdivided resources into a foreign domain and/or be hosted on a different (e.g., deployed) network.

3.2.1.2.4 Dynamic Addressing. The implementation shall support configurations where hosts [including hosts that provide distributed computing servers?] in the domain are frequently moved and must determine their network IP address at boot-up using the technique known as dynamic IP addressing.

3.2.1.2.5 Mobile Operation. [TBD]. It has yet to be determined if a requirement exists for the ability for the distributed computing implementation to support operation while moving (e.g., like cell phone operation).

3.2.1.3 Synchronized Time

3.2.1.3.1 Time Service. The implementation shall provide for automatic, secure, global synchronized time.

3.2.1.4 Performance and Architecture

3.2.1.4.1 Scalability

NOTE: in the following requirements, rough, order-of-magnitude estimates of maximums have been provided. These are highly subjective. The range of scalability is a factor that should be considered as part of the COE product evaluation/recommendation cycle.

3.2.1.4.1.1 Intra/Inter-net/Remote Network Scalability. The implementation shall support scalable operation over intra-, internet, and remotely connected networks, proportionate to network speed and available capacity. At the low end, for remotely connected hosts, a minimum of 9.6Kb/s line speed with varying available capacity should be assumed.

3.2.1.4.1.2 Domain Scalability. The implementation shall provide scalable performance as the number of domains (e.g., cells, namespaces) increases. A maximum of domains numbering in the low thousands should be assumed.

3.2.1.4.1.3 Usage Scalability. The implementation shall provide scalable performance as the number of users/principals/servers/objects in a domain increases. Maximum site sizes may have up to 50,000 users, numbers of servers in the low hundreds, and numbers of objects in the several thousands.

NOTE: objects can exist at varying levels of granularity, and it is not difficult to imagine hundreds of millions of objects. However, it is likely that the level of granularity utilized in the COE will require more than several thousand in the COE V4-V5 timeframe.

3.2.1.4.1.4 Server Load Balancing. The implementation shall provide the ability to distribute client requests amongst replicated servers to facilitate optimal performance by exploiting any parallelism that may exist in the configuration of servers.

NOTE: DCE binding APIs provide support for the client to select from multiple available servers on either a random basis or on the basis of some other criteria that the client applies. Products like Encina extend this to provide for bindings that distribute client requests based on server load.

3.2.1.4.2 Deleted (this paragraph should be removed from the document prior to finalization).

3.2.1.4.3 Concurrency/Threading. The implementation shall provide concurrent access to services and shall support multithreading of service implementations.

3.2.1.5 Fault Tolerance

3.2.1.5.1.1 Replication. The implementation shall, in the event of server failure, support: a) automatic rebinding of clients to replicated servers, where those servers are a part of the distributed computing component (e.g., DCE CDS, Security Server, etc, and b) graceful notification to the client of a server failure with the opportunity for the client to rebind to a replicated server if appropriate for the application.

3.2.1.5.1.2 Server Failure Management. The implementation shall provide the capability of monitoring the health of the servers, and be capable of automatically restarting failed servers.

3.2.1.5.1.3 Reliability. The implementation shall provide the ability to guarantee that requests for services are implemented reliably, such that the requestor can know deterministically whether the request was performed by the server.

3.2.1.6 Language Support

3.2.1.6.1 Ada, C, C++. The implementation shall provide support for application software development in the following programming languages: Ada'95 (including Ada Task compatibility with threading in the distributed computing component and in the operating system), ANSI C, C++.

NOTE: We may need to specify compiler products for Ada'95 and C++ since these languages are currently either incompletely implemented or not described by formal specifications, respectively.

NOTE: Java requirements are included in the CORBA Specific Requirements and DCE Specific Requirements sections, later.

3.2.1.6.2 Reserved.

3.2.1.6.3 Compiler Support. The implementation shall support software development using the native compiler for each COE supported platform.

3.2.1.7 Transaction Processing

3.2.1.7.1 Atomicity. The implementation shall provide support of atomicity for ensuring that a computation consisting of one or more operations on one or more objects satisfies the requirements of atomicity (if a transaction is interrupted by a failure, any partially completed results are undone).

3.2.1.7.2 Isolation. The implementation shall provide the ability of transactions to execute concurrently, with the same result as if they were performed sequentially.

3.2.1.7.3 Durability. The implementation shall provide support for durability (if a transaction completes successfully, the results of its operations are never lost, except in the event of catastrophe).

3.2.1.7.4 Database Support. The implementation shall provide support for 3-tier applications including support for multiple databases (same or different database vendors) and multiple platforms including all of the COE platforms and database management systems.

3.2.1.7.5 Database Management Heterogeneity. The implementation shall support transactions that span across the breadth of the multiple, heterogenous database management systems that are supported by the COE (e.g. Oracle, Sybase...).

3.2.1.7.6 Process Spanning Transactions. The implementation shall provide the ability to have transactions span across multiple processes (e.g. Process A starts a transaction, Process B continues the transaction, Process A completes the transaction).

3.2.1.7.7 Database Independent API. The implementation shall provide an transaction processing API that is independent of the database management system.

3.2.1.7.8 Transaction Rollback. The implementation shall provide the ability to abort transactions and cause all involved databases to rollback to their initial state (before transaction began).

3.2.1.7.9 Nested Transactions. The implementation shall provide the ability to nested transactions.

3.2.1.8 Queueing

3.2.1.8.1 Persistence. The implementation shall maintain an object in a queue until it has been de-queued, and shall provide for reliable recovery of queue contents in the event of a system restart or failure.

3.2.1.8.2 Queue Query. The implementation shall provide the capability to access (read) queued objects while they are still in the queue (i.e. you should not have to de-queue an object before you can read it).

NOTE: Current application level uses of queueing functionality require performance of at least 30 operations per second. Because performance is relative to a wide variety of factors unrelated to queueing, performance should be part of product evaluation criteria, but is probably inappropriate to define as a requirement.

3.2.1.8.3 Deleted. (This paragraph should be removed before this document is finalized).

3.2.1.8.4 Priorities. The queue service must support queuing and de-queuing of objects at at least 10 different priority levels.

3.2.1.8.5 Queue Polling. The implementation shall support: a) Synchronous Blocking, where a queue is polled and the process is blocked until something arrives in the queue; b) Synchronous Non-Blocking, where a queue is polled and control is immediately returned to the process whether there are any objects in the queue or not; and c) Asynchronous, where a queue is polled, control is immediately returned to the process, but the process is notified (at some later time) when an object arrives on the queue.

3.2.1.8.6 Queue Size. The implementation shall be capable of accomodating queue objects of at least 50K bytes.

3.2.1.8.7 Number of Queues. The implementation shall be able to create and maintain a minimum of 100 (simultaneous) queues.

3.2.1.8.8 Concurrent access. The implementation shall support: a) concurrent access to queues, b) simultaneous queueing from multiple processes, and c) multiple processes reading from the same queue.

3.2.1.8.9 Multiple Queues. The implementation shall support the ability for a process to have multiple (incoming) queues.

3.2.1.8.10 Queued Object Size. The implementation shall be capable of accommodating queue objects of a least 50K bytes.

3.2.1.8.11 Access Control. The implementation shall support the ability to define access control lists on a per queue basis.

3.2.1.8.12 Queue Locking. The implementation shall support the ability to lock queue entries to protect against concurrent write access.

3.2.1.8.13 LIFO. The implementation shall provide the ability to access the queue in last-in-first-out (LIFO) order.

NOTE: The recommended solution from Transarc does not satisfy the above requirement for LIFO access.

3.2.1.9 Management

3.2.1.9.1 License Management. The implementation shall provide tools for managing any licensing mechanisms that are required by the implementation. These tools should preferably be GUI based, and should be easy to use by systems administrators. Any license management should be integrated with the othe relevant management functions.

3.2.1.9.2 Transaction Management. The implementation shall provide an administrative application to monitor transactions, including performance, failed transactions, and status of servers, and to start/stop the transaction processing monitor.

3.2.1.9.3 Reserved.

3.2.1.9.4 Queue Management. The implementation shall provide an administrative application to monitor queues, providing the following capabilities:

a) Report the number of objects in each queue.

b) Report all processes connected to each queue (process name and machine it is running on).

c) Report the status of each process (e.g. waiting, reading, writing)

d) Flush any or all queues.

e) Start and stop the queue service servers.

3.2.1.9.5 Reserved.

3.2.1.9.6 Reserved.

3.2.1.10 Reserved

3.2.1.11 Reserved

3.2.1.12 Reserved

3.2.1.13 Reserved

3.2.1.14 Reserved

3.2.1.15 Reserved

3.2.1.16 Documentation

3.2.1.16.1 Developer Documentation. The implementation shall include documentation to describe the proper or recommended usage of the implementation by software developers, as well as explicitly identify usage which is prohibited by the architectural tenets of the COE or which would be incompatible with other COE capabilities.

3.2.1.17 Y2K Compliance

3.2.1.17.1 Y2K Compliance. The implementation shall be Y2K compliant, as warranteed by a letter from each product vendor.

3.2.2 DCE Specific Requirements

To address distributed computing for the remote procedural call based software development paradigm, the DII has adopted the Distributed Computing Environment (DCE) technology, defined by the Open Group (previously the Open Software Foundation). There are many reasons why DCE was selected, most of which are beyond the scope of this SRS. The DII COE specific requirements for the use of DCE are specified in the next sections.

3.2.2.1 DCE Version

3.2.2.1.1 Version Compliance. The implementation shall be compliant with OSF DCE V1.2.x.

3.2.2.2 DCE services

3.2.2.2.1 Theads. The implementation shall use the native operating system threads services. If the operating system implementation of threads is unsupported, the DCE implementation shall provide an implementation of the POSIX pthreads services.

3.2.2.2.2 Naming. The implementation shall provide the DCE Cell Directory Service and utilize the DNS or LDAP directory service for locating other cells.

NOTE: X.500 based Global Directory Services (GDS) are not subject to widespread commercial availability, due in part to the nearly ubiquitous deployment of DNS in the commercial and DoD communities. However, the COE may be required to support X.500 based GDS in the future to align with other major government initiatives such as the Defense Message System which is utilizing X.500.

3.2.2.2.3 Security. The implementation shall provide the DCE Security Service.

3.2.2.2.4 Reserved.

3.2.2.2.5 Transitive trust. The implementation shall provide transitive trust between hierarchical cells, such that principals may access services located in other cells without the need for pair-wise registration of principals between cells.

NOTE: The currently recommended DCE product from Transarc does not satisfy the transitive trust requirement. Transitive trust has not yet been implemented by the Open Group; This situation should be remedied by the COE V3.3 timeframe.

3.2.2.2.6 Reserved.

3.2.2.2.7 Reserved.

3.2.2.2.8 Time. The implementation shall provide a DCE Distributed Time Service (DTS) that is capable of interfacing with NTP for time synchronization outside of the cell.

3.2.2.2.9 Host. The implementation shall provide the DCE Host services.

3.2.2.3 DCE Applications

3.2.2.3.1 Distributed File System: The implementation shall provide the DCE Distributed File System (DFS), including at least the DFS Client and DFS Exporter functionality.

3.2.2.3.2 NFS/DFS Gateway. The implementation shall provide a gateway that permits hosts to use NFS to access files in the DFS.

3.2.2.4 DCE Software Development

3.2.2.4.1 Reserved.

3.2.2.4.2 Application programming interface: The implementation shall provide the API implementation of the standard DCE API and Generic Security Service API (GSS-API).

3.2.2.4.3 Reserved.

3.2.2.4.4 Reserved.

3.2.2.4.5 Reserved.

3.2.2.4.6 Reserved.

3.2.2.4.7 Ease of use APIs. The implementation shall provide an API that provides a more abstract, higher level interface to the common DCE programming idioms, such as client and server registration and initialization, obtaining a server binding, performing name/directory service lookups, use of security services (including ACL access, monitoring, and auditing), use of the GSS-API, use of RPCs, and access to identification and authentication information.

3.2.2.5 Management

3.2.2.5.1 Cell Management. The implementation shall provide GUI based tools to support both local and remote (but within the cell) configuration or reconfiguration of DCE services, including support for:

a) Add/delete a host from a DCE cell.

b) Relocate a host from one cell to another.

c) Attaching a host to another network.

d) Maintenance of dynamic IP addressing capabilities.

e) Start/Stop servers.

f) Add/Delete/Modify and configure servers in a cell.

g) Maintenance of cell and lan profiles.

h) Maintaince of the endpoint map of hosts.

i) Monitor server status and be capable of starting or restarting servers upon reboot, failure, or as part of normal operating procedures.

j) Maintain the security server, including the registry, groups, access control lists, and all attributes associated with each.

k) Synchronize the security registry contents with the related operating system information, including user account and group information.

l) Maintain the time server.

m) Add/Delete/Modify filesets in the distributed file system.

n) Synchronize master and replicated servers.

o) Rebuild master servers after failure.

p) Relocate servers to different hosts.

q) Create and maintain DCE server replicas, including the security server, time server, DFS filesystem servers, and cell directory server

r) Create and maintain the hierarchical Cell Directory Service, including all of the attributes associated with the CDS.

s) Remote management of the implementation.

t) Maintain the audit functionality, including the collection, synchronization, reduction, and archival of audit logs and the start/stop of the DCE audit daemons on hosts.

u) Register cells for inter-cell authentication.

v) Backup and restore DCE server data, including at least the CDS directory and security server data.

w) Browse and search the CDS namespace.

3.2.2.6 Compatibility and Migration Support

3.2.2.6.1 3-Tier Migration. The implementation shall provide tools to ease migration of legacy and 2-tier applications to a 3-tier architecture.

3.2.2.6.2 Network Protocols. The implementation shall provide the ability to add DCE's security functionality to the following network protocols, such that the protocols exhibit the security benefits of DCE and remain compatible with operating system supplied versions of these same protocols:

a) Remote Shell (rsh/rshd)

b) Remote Execution (rexec/rexecd)

c) Remote Login (rlogin,rlogind)

d) Remote Copy (rcp, rcpd)

e) Telnet
f) FTP
g) SMTP
h) SNMP
i) HTTP

3.2.2.6.3 Reserved.

3.2.2.7 DCE Default Configuration

The implementation shall provide a default configuration for the DCE Cell Directory Service (including namespace configuration in accordance with the DII COE DCE Implementation Plan) and DCE Security Server (including default principals and configuration to implement DII security policies).

3.2.2.8 DCE Documentation

3.2.2.8.1 Concept of Operation. The implementation shall provide a DII COE DCE Concept of Operations (CONOPS) document describing the overall definition, role, operation and maintenance of DCE cells in the DII. This is a higher level document than the DII DCE Adminstration Guide.

3.2.2.8.2 Administration Guide. The implementation shall provide a DII COE DCE Administration Guide that describes procedures and tools for the details of daily administration of a DCE cell and how to interconnect DCE cells within and across, organizational boundaries.

3.2.2.9 Java Language Support

3.2.2.9.1 (Future Requirement) Access to DCE Services using Java. The implementation shall provide the capability to access distributed DCE based services from Java applets and applications.

3.2.2.10 Microsoft NT Support

3.2.2.10.1 Access to DCE Servers. The implementation shall provide the capability to access distributed DCE based services from Windows NT clients.

3.2.2.11 Naming and Address Control

3.2.2.11.1 Workstation Directory. Provide the capability to create and maintain addressing directories that contain information about all distant W/Ss, regardless of cell, that are operationally scheduled to be in communications with the local W/S.

3.2.3 CORBA Specific Requirements

To address distributed computing needs for the object-oriented software development paradigm, the DII has adopted the Common Object Request Broker (CORBA) technology, defined by the Object Management Group (OMG). There are many reasons why CORBA was selected, most of which are beyond the scope of this SRS. The DII COE specific requirements for the use of CORBA are specified in the next sections.

3.2.3.1 Reserved ~~(Background)~~

3.2.3.2 CORBA Version

The implementation shall be compliant with version 2.0 of the CORBA specifications, as specified by the Object Management Group.

Note: There is not currently a validation and compliance testing suite, so compliance right now is not something that can easily be verified. Check the CORVAL page on the web for the status of CORBA compliance testing.

3.2.3.3 CORBA Interfaces

3.2.3.3.1 CORBA. The implementation shall provide implementations of the following adopted CORBA interfaces:

a) ORB core

b) IIOP

c) Implementation Repository

d) Interface Repository

e) IDL compiler

f) Static Invocation Interface

g) Dynamic Invocation Interface

h) Dynamic Skeleton Interface.

3.2.3.3.2 CORBAservices. The implementation shall provide the following CORBAservices as defined by the OMG:

a) Naming

b) Event Management

c) Transaction

d) Lifecycle

e) Security

f) Query

g) Time

Note: Some of the CORBAservices specified above have not yet been implemented by vendors, although they have been adopted by the OMG. Those that are specified for COE V4.0 are expected to be available within the needed time frame.

3.2.3.3.3 Future CORBA Services. In the future, the implementation shall provide the following additional CORBAservices:

a) Concurrency

b) Relationship

c) Licensing

d) Persistence

e) Trader

f) Properties

g) Externalization.

3.2.3.4 CORBAfacilities

3.2.3.4.1 CORBAfacilities. The implementation shall provide the following CORBAfacilities as specified by the OMG: a) Compound Document Presentation and Data Interchange.

Note: This facility, as currently specified by OMG, is based on OpenDoc, which became obsolete technology in the spring of 1997 when the principal supporting vendors withdrew longer term development and support.

3.2.3.5 CORBA Applications

3.2.3.5.1 Interface Repository Browser: The implementation shall provide a GUI-based capability for browsing the interfaces that are contained in the interface repositories of local and remote systems, as permitted by security policy.

3.2.3.6 Reserved

3.2.3.7 Management

3.2.3.7.1 Implementation repository management: The implementation shall provide a GUI-based tool for managing the contents and configuration of local and remote implementation repositories.

3.2.3.7.2 Interface repository management: The implementation shall provide a GUI-based tool for managing the contents and configuration of local and remote interface repositories.

3.2.3.7.3 Namespace management: The implementation shall provide a GUI-based tool for managing the contents and configuration of the CORBA namespace.

3.2.3.7.4 Security management: The implementation shall provide a GUI-based tool for managing the security configuration of the CORBA implementation.

3.2.3.8 Compatibility and Migration Support

3.2.3.8.1 DCE Compatability. The implementation shall be compatible with the COE implementation of the DCE.

NOTE: To the extent possible, the CORBA and DCE implementations should leverage off of each other's strengths, and CORBA capabilities should re-use or be layered upon the DCE implementation such that duplication is minimized and greater consistency is obtained.

3.2.3.8.2 Application Service Wrapping. The implementation shall provide the capability to access DCE enabled application services.

Note: The above might take the form of a CORBA/DCE generic bridge, or might involve the wrapping of DCE application services with CORBA wrappers. The practical ability to accomplish this will probably have to be determined on a case-by-case basis, depending on the DCE services that the application services uses, such as DCE pipes and pointers.

3.2.3.8.3 Microsoft Distributed Common Object Model (DCOM). The implementation shall the capability for OLE objects to request services from CORBA objects and vice-versa, using CORBA adopted technology.

3.2.3.9 Java Language Support

3.2.3.9.1 Access to CORBA Services using Java. The implementation shall provide the capability for Java Bytecode applets and applications to access distributed CORBA based services.

3.2.3.9.2 Java Servers. The implementation shall provide the capability to implement CORBA servers using the Java language, and to access to such distributed CORBA services from Java and non-Java clients.

3.2.3.10 Microsoft NT Support

3.2.3.10.1 Access to CORBA Servers from NT. The implementation shall provide the capability to access distributed CORBA based services from Windows NT clients.

3.3 CSCI EXTERNAL INTERFACE REQUIREMENTS

3.3.1 Interface identification and diagrams

3.3.2 Project-unique identifier of interface

3.4 CSCI INTERNAL INTERFACE REQUIREMENTS

3.5 CSCI INTERNAL DATA REQUIREMENTS

3.6 ADAPTATION REQUIREMENTS

3.7 SAFETY REQUIREMENTS

3.8 SECURITY AND PRIVACY REQUIREMENTS

Specified earlier.

3.9 CSCI ENVIRONMENT REQUIREMENTS

3.9.1 Platform Requirements

The implementation shall support the platforms (hardware/operating system combinations) specified for the DII COE.

3.9.2 Network Requirements

The implementation shall support the network specified for the DII COE.

3.10 COMPUTER RESOURCE REQUIREMENTS

3.10.1 Computer hardware requirements

3.10.2 Computer hardware resource utilization requirements

3.10.3 Computer software requirements

3.10.4 Computer communications requirements

3.11 SOFTWARE QUALITY FACTORS

3.12 DESIGN AND IMPLEMENTATION CONSTRAINTS

3.13 PERSONNEL-RELATED REQUIREMENTS

3.14 TRAINING-RELATED REQUIREMENTS

3.14.1 Product training

Product Training. The implementation shall provide on-site training for each of the COTS and GOTS products comprising the implementation, requisite with the use of the products.

3.14.2 DII COE Training

The implementation shall provide centralized training that supplements the product training described above, and that is tailored to provide instruction on how to use the implementation within the DII COE context, including:

3.14.2.1 Installation Training

The implementation shall provide training that prepares the student for performing the installation of all of the components of the implementation.

3.14.2.2 System Management Training

The implementation shall provide training that prepares the student to management the implementation, including procedures that may be unique to the DII COE context, such as security.

3.14.2.3 Software Development Training

The implementation shall provide training that prepares the student for software development using the allowed features of the implementation, including the use of programming idioms, templates, testing, or other methods that may be unique to the DII COE context.

3.15 LOGISTICS-RELATED REQUIREMENTS

3.16 OTHER REQUIREMENTS

3.17 PACKAGING REQUIREMENTS

3.18 PRECEDENCE AND CRITICALITY OF REQUIREMENTS

SECTION 4

QUALIFICATION PROVISIONS

TBD.

SECTION 5

REQUIREMENTS TRACEABILITY

This section under separate enclosure.

SECTION 6

NOTES

None.

APPENDIX A

WORKING GROUP PRODUCT RECOMMENDATIONS

7.1 RECOMMENDATIONS TO THE DII COE ARCHITECTURE OVERSIGHT BOARD

7.1.1 Transarc DCE

The Distributed Computing Environment (DCE) version 1.1 product from Transarc is recommended to satisfy all of the core DCE requirements, including the implementation of time, threads, directory services, RPC, and security. Additionally, Transarc's implementation of the DCE Distributed File System (DFS) and NFS/DFS gateway is recommended for sites that require such functionality. Transarc's implementation of DCE, however, does not currently support Ada'95 language bindings or mobile hosts, which are identified as COE requirements. Transarc's DCE is available on all of the COE platforms.

Another, government-owned implementation of DCE version 1.1 was developed by the Army. The Army's implementation, contracted to Unixpros, uses the same source code baseline as the Transarc DCE product, with modifications to support both the Ada'95 bindings and mobile host requirements that were identified above as deficiencies in the ability of the Transarc implementation to satisfy COE requirements. Conversely, though, there are are limitations regarding the availability of the Army's implementation on the range of COE platforms that are required.

The decision to recommend Transarc's DCE vice the Army/Unixpros solution hinged on three factors: 1) COTS vs GOTS. The recommendation is consistent with DOD direction to use COTS, and addresses requirements for best commercial practices with regard to training, documentation, technical support, and availability. 2) Cost. The recommendation is believed to be in the best cost interests of the majority of the COE users. There are, however, implementation scenarios in which deployment costs (not including support, maintenance, training, etc) could exceed the alternative Army/Unixpros solution. 3) Availability and porting. The recommendation is available on the full range of COE platforms required, as well as other platforms that are not yet required but which may be in the future.

7.1.2 Open Horizons Connection [with qualifications]

The Connection product from Open Horizons supports migration of legacy applications to DCE by providing DCE'ized versions of runtime libraries for other COTS products, principally Oracle. The Connection product, however, has some limitations in that it supports only a subset of the Oracle database APIs that are in use in legacy systems. Hence, the product may also be limited in its use in conjunction with 4GL database forms packages, such as Sybase's Gain Momentum, and so it's recommendation must be qualified to consider the case-by-case applicability of the tool.

Note: Requirements for DCE'izing legacy database connections are somewhat weak, and are generally subsumed under the category of migration requirements.

Note: The working group is investigating other alternatives that may provide similar functionality, such as Intellisoft's DCE/Snare.

7.1.3 HAL DCE Cell Manager

The DCE Cell Manager product from HAL provides a GUI-based interface to most DCE cell management functions. It is being upgraded to support DCE version 1.1, including hierarchical cells and access control list management. The product, however, is not integrated with the products being used for COE Management Services, and the ability of the product to support the CONOPS for COE system administration and security has not been evaluated.

7.1.4 Transarc Encina (including the Recoverable Queueing Service)

Encina/Encina++ from Transarc provides a higher level API to the DCE functionality, and provides support for the reliability requirements that are inherent to transaction processing. Encina++, included with Encina, provides an object oriented API to the Encina functionality, with bindings to the C++ language. Encina satisfies replication and load balancing requirements without additional application programming, and satisfies the monitoring requirements. The Reliable Queueing Service, an optional product in the Encina family, satisfies all but the Army's LIFO requirements for queueing and is also recommended to satisfy the queueing requirements. Both products rely upon DCE core services.

7.1.5 CORBA ~~(TBD)~~

~~CORBA recommendations are TBD.~~ The DISA/DARPA JPO has segmented Iona’s Orbix for C++ V2.1c on Solaris 2.5.1. As of 7/97, no other CORBA products had been nominated, although the Visigenic Visibroker for Java ORB is bundled by Netscape into Navigator V4.0 which is defined to be a part of the COE.

7.1.6 CORBA/Ada Mappings ~~(TBD)~~

~~CORBA recommendations are TBD.~~ As of 7/97, the only CORBA product to be segmented is Orbix for C++ on Solaris. There are only two known products that implement the Ada’95 Language Mapping for CORBA; It is recommended that the Orbix/Ada product from Objective Interface Systems be used to provide the Ada Language Mapping for Orbix. This requires the development of another segment, since the product is substantially different than Orbix for C++. No Ada'83 product is known to exist.

APPENDIX B

REQUIREMENTS FOR OTHER COE COMPONENTS

8.1 OPERATING SYSTEM REQUIREMENTS

8.1.1 Time

System time coordination with DCE/CORBA time services.

8.2 MANAGEMENT SERVICES

8.2.1 Common Desktop Environment

8.2.1.1 Single Login Integration

8.2.1.1.1 Single Login Integration. The implementation shall be integrated with the DII COE console login facilities, defined by the Management Services implementation, such that executation of the normal console login sequence identifies and authenticates the user, allowing the user with DCE-enabled capabilities. Currently, this requires integration of DCE with the Triteal Enterprise Desktop product.

8.2.1.1.2 DFS/CDE integration. The implementation of DCE/DFS shall be integrated with the Triteal Enterprise Desktop product.

8.2.1.1.3 Remote execution integration. The implementation shall be integrated with the Triteal Enterprise Desktop product to allow desktop startup of remote processes using the facilities provided by DCE.

8.2.1.1.4 The single login capability shall continue for the duration of a user session, until the user logs out. Requirement for IOC and will require adding in Access Manager in front of COE solution (DCE?, Tivoli?) Product of choice: Access Manager [Needed in COE 3.2 on Win/NT and UNIX for IOC at JAX, priority HIGH, per Ms Lorna Estep, Logistics Community Manager in COE 3.2 requirements call].

8.2.1.1.5

8.2.2 System Management

8.2.2.1 DCE/System Management Integration

8.2.2.1.1 DCE/System Management Integration. The implementation of the cell management capabilities shall be integrated with the UNIX user and security management capabilities, specified in the DII COE Management Services SRS.

8.2.2.1.2 CORBA/System Management Integration. The implementation of the CORBA management capabilities shall be integrated with the UNIX system management capabilities, specified in the DII COE management Services SRS.

8.3 COMMON SUPPORT APPLICATIONS

8.3.1 Netscape/Mosaic

DCE/WEB integration.

8.3.2 Java

DCE/IDL mappings.

8.4 SOFTWARE DEVELOPMENT SERVICES

8.4.1 Design

8.4.1.1 Object Oriented Analysis and Design

8.4.1.1.1 Object Oriented Analysis and Design Tools: The implementation shall provide GUI-based tools for performing object oriented design and analysis as part of the software development environment.

8.4.2 Testing

8.4.2.1 Automatic test generation tools.

8.4.2.1.1 Automatic Test Generation Tools. The implementation shall provide tools to support the automatic generation of tests.

8.4.3 Development

8.4.3.1 DCE Software Development

8.4.3.1.1 Wrappers. The implementation shall provide example, or template, DCE wrappers that show how to encapsulate an existing non-DCE executable application such that it can be invoked via DCE, including wrapper backends that perform:

a) Command line based service invocation.

b) Other (TBD).

8.4.3.1.2 DCE traffic monitor/debugger: The implementation shall provide a GUI-based tool for monitoring DCE traffic between clients and servers that can be used to assist in debugging the clients, servers, and DCE configuration.

8.4.3.1.3 Templates: The implementation shall provide example client and server software templates that demonstrate typical usage of the DCE capabilities, for each of the supported programming languages.

8.4.3.2 CORBA Software Development

8.4.3.2.1 Inter-ORB traffic monitor/debugger: The implementation shall provide a GUI-based tool for monitoring CORBA traffic between clients and servers that can be used to assist in debugging the clients, servers, and CORBA configuration.

8.4.3.2.2 Templates: The implementation shall provide example client and server software templates that demonstrate typical usage of the common CORBA interfaces, for each of the supported programming languages.

8.5 GENERAL SYSTEM ENGINEERING

8.5.1 Cost

- - Licensing agreements for all GCCS Distributed Computing software on a per-site basis. Per-site licensing (as opposed to per-machine licensing) allows sites to have greater flexibility in configuring and maintaining their system. Maintaining licenses for each machine is more time consuming and places an additional burden on support personnel.

- - One negotiated price (by DISA) for all of the services. JMCIS sites would then be responsible for purchasing licenses, based on the negotiated price.

- - Each JMCIS site must have the capability to run multiple server machines. To take full advantage of distributed computing features like 3-tier architecture, server replication, and load balancing JMCIS sites will require the ability to install GCCS Distributed Computing software on several servers and clients. Licensing and pricing must permit this type of flexibility.

Tools must be cost effective and support required platforms.

8.5.2 Documentation

8.5.2.1 DII COE DCE implementation plan

The implementation shall include a DII COE DCE Implementation Plan document. This is primarily a planning document, and shall describe the implementation in suffucient detail to be able to guide planning efforts as well as assist in determining the work plan for the DCWG and DISA JIEO activities related to procedural based distributed computing.

8.5.2.2 DII COE DCE application programmer’s guidance

The implementation shall include a DII COE DCE Application Programmer's Guidance document. This document shall describe the proper or recommended usage of the implementation by software developers, as well as explicitly identify usage which is prohibited.

8.5.2.3 DII COE CORBA implementation plan

The implementation shall include a DII COE CORBA Implementation Plan document. This is primarily a planning document, and shall describe the implementation in suffucient detail to be able to guide planning efforts as well as assist in determining the work plan for the DCWG and DISA JIEO activities related to object-oriented distributed computing.

8.5.2.4 DII COE CORBA application programmer’s guidance

The implementation shall include a DII COE CORBA Application Programmer's Guidance document. This document shall describe the proper or recommended usage of the implementation by software developers, as well as explicitly identify usage which is prohibited.

8.5.3 COE Services

8.5.3.1 DCE/CORBA Migration

DII COE services that are implemented as DCE-based application services shall also be accessible via a CORBA interface.

8.6 NETWORK SERVICES

The following requirement is also listed in section 3.2.1.1.3. See explanatory note below.

8.6.1.1.1 Firewalls. The implementation shall provide support for the use of DCE services through firewalls and guards.

NOTE: Operation of DCE through firewalls and guards is probably not a requirement that can be directly satisfied by the distributed computing component of the COE, but is more likely to be satisfied through the configuration of the entire system/network, including routers, packet filtering, intermediate hosts, etc. Even so, the implementation of distributed computing shouldn't deny service in such configurations.

8.7 DATA ACCESS SERVICES

8.7.1.1.1 The data access service shall support access to data objects using CORBA standards. [Per Ms Lorna Estep, Logistics Community Manager,

8.7.1.1.2 The Data Access Agent shall provide CORBA-based services for common data entities (i.e. objects) and common functions. These services shall provide for databse data dn business rules abstractions for objects such as targets, airspaces, enemy order of battle. [Per Iraj Bighash, Lockheed Martin/TBMCS, for COE 3.1 and 3.2, platforms not indicated]

8.7.1.1.3 RDBMS shall provide DCE Support (e.g., Oracle Advanced Networking option, Informix DCE/NET) [Per Clarissa Reberkenny (Director, DMIM/TI&S), Health Affairs, for MHSS, on COE 3.2, on NT 4.0, as priority “essential”].

8.8 OFFICE AUTOMATION

8.8.1.1.1 Web Server shall provide an CORBA 2.0 compliant ORB. [Per Clarissa Reberkenny (Director, DMIM/TI&S), Health Affairs, for MHSS, on COE 3.2, on NT 4.0, as priority “desirable”].

8.8.1.1.2 Support for DCE user authentication (for news readers). . [Per Clarissa Reberkenny (Director, DMIM/TI&S), Health Affairs, for MHSS, on COE 3.2, on NT 4.0, as priority “desirable”].

APPENDIX C

MISCELLANEOUS EVALUATION CRITERA

9.1.1.1.1 Legacy Compatibility. The implementation shall be consistent with the requirements of legacy and migration systems that will utilize the COE.

9.1.1.1.2 Product Quality. The implementation shall be of a quality consistent with the best commercial practices of the industry, including continuing product improvement, bug fixes, telephone and email support, documentation, interoperability, and training.